SEMICONDUCTOR DEVICE TEST APPARATUS AND METHOD

TECHNICAL FIELD

The present invention relates to a semiconductor device test apparatus and method which replaces defective memory cells of a memory of a semiconductor device with spare memory cells provided beforehand, thereby repairing the defective semiconductor device.

BACKGROUND ART

An apparatus for analyzing defective memory cells of a memory of semiconductor devices fabricated as integrated circuits to determine a way to efficiently repair the defective memory cells using spare memory cells is called the MRA (Memory Repair Analyzer). The semiconductor devices are not limited to dedicated memory devices; they may be any semiconductor devices containing memory devices. Semiconductor devices to which the present invention is applied will be hereinafter referred to as memory devices and a memory device to be tested will be referred to as a DUT (Device Under Test). There are a number of MRA approaches. One typical example will be described below.

Fig. 8A shows a wafer 110 on which a large number of semiconductor devices are formed in an array. MRA is performed in the stage preceding a memory test, that is, it is performed in an on-wafer stage. A large number of memory devices are formed in an array on the wafer 110. Each of the memory devices has multiple memory blocks, each consisting of many memory cells. As a memory block 120 is schematically shown in Fig. 8B, more than one spare lines 130, 135 are provided for each memory block 120 along the rows (in the X-axis direction) and the columns (in the Y-axis

direction).

Here, suppose that a test on the memory block 120 revealed that memory cells 122, 125, and 127 are defective. The test result data can be analyzed using MRA to determine a repair solution. Fig. 8C shows a repair solution in which the column line 131 of a memory cell 122 is replaced with one of the column spare lines 130 and the row line 136 of memory cells 125 and 127 is replaced with one of the row spare lines 135 by the MRA. Such repair of defective cells of memory devices formed on a wafer is described in United State Patents No. 6345004 and No. 6243307, for example.

The MRA is means capable of analyzing fast which spare line a defective memory cell found in a memory device can be replaced with to repair the defect, by using dedicated hardware and software. The MRA has been sufficiently effective for repairing conventional memory ICs.

Memory devices have made remarkable progress in recent years. With the advent of 64M-SDRAM, more and more memory devices have unique redundancy structures specified by users (LSI manufactures) and have become complicated. As a result, a problem has arisen that the result of MRA analysis must be adjusted through a post process (post process at an on-wafer stage) as will be described later. That is, the conventional MRA capabilities are used to obtain a repair solution and then an engineering work station (EWS) must be used in a post process to make adjustments to the solution.

However, because this method can adjust only one solution determined by the MRA, it has been impossible for a post process to determine whether there is a solution that makes a defect found to be unrepairable as a result of adjustments in the post process repairable. Consequently, the yields have been reduced. An example of a device that

could not be repaired using conventional MRA because of a redundancy structure will be described below.

The memory device shown in Fig. 9 consists of four block groups BG1 - BG4. Block groups BG1 and BG2 make up BANK-A and block groups BG3 and BG4 make up BANK-B. Each block group consists of four blocks. Two spare row lines 135 are provided in each bank for repairing two block groups at a time. Two column spare lines 130 are provided for each block. The following three conditions are required to assign spare lines for repair of a defective memory cell.

Fig. 10 illustrates a first condition. The two spare lines in each block can repair any defective cells in the same block group freely in principle, with some exceptions.

Fig. 11 illustrates a second condition. The second condition is an exception to the first condition. The spare line of the rightmost block BL4 in the same block group cannot repair defective cells in the leftmost block BL1. Similarly, the spare lines of the leftmost block BL1 cannot repair defective cells in the rightmost block BL4.

Fig. 12 illustrates a third condition. This condition is as follows. Consider two adjacent blocks in two adjacent block groups, for example block BL4 in block group BG3 and block BL5 in block group BG4, in the same bank. If fails at address "a" that have occurred in block BL4 are repaired with a spare line of block BL4, fails at the same address "a" in the adjacent block BL5 cannot be repaired with a spare line of block BL5. However, address "a" in block BL5 can be repaired with a spare line of adjacent block BL6.

The example shown in Fig. 9 is of a memory device having three constrains of a redundancy structure. There are many other memory devices

having various structures. It has become impossible to repair all of the various types of memory devices with conventional MRA. For example, the conventional MRA can address the first and second conditions in the memory device in Fig. 9, but not the third condition.

As has been stated above, the first and second conditions in the memory device having the redundancy structure shown in Fig. 9 can be addressed with the conventional MRA. After the MRA is performed, a post process must be performed for checking the third condition on the EWS to adjust the result of the repair. In addition, since only one repair solution of the first and second conditions is obtained before the third condition is checked, an optimum solution cannot necessarily be obtained and, consequently, a reduction in the yield can result.

That is, for the example described above, the conventional MRA requires, as the conceptual diagram shown in Fig. 13, performing a function test on a DUT (step S140), performing memory repair analysis (MRA) by inputting data on the results of the test to obtain a repair solution under the restrictions of the first and second conditions (step S141), storing temporarily the repair solution (step S142), and making adjustments to the result and the third condition on the EWS (step S143) to obtain a final repair solution.

An object of the present invention is to provide a semiconductor device test apparatus and method capable of analyzing and repairing the first, second, and third conditions at a time using a general-purpose MRA.

DISCLOSURE OF THE INVENTION

A semiconductor device test apparatus according to the present invention includes:

a test processor which applies a test signal to a semiconductor device

under test and obtains information about a defective memory cell from a response signal; and

a repair analysis computing unit which performs repair analysis of the defective memory cell information to determine a way to repair the defective memory;

wherein the repair analysis computing unit comprises:

memory repair analysis means for performing repair analysis of the defective memory cell information in accordance with a memory repair analysis program and determining assignment of a spare line to the defective memory cell; and

user function means for inserting a user function based on a user-specified user repair analysis program between desired units of processing of the memory repair analysis program to make a change to data processed by the memory repair analysis program.

A semiconductor device test method according to the present invention includes the steps of:

(a) performing a function test on a memory of a semiconductor device under test to obtain information about a defective memory cell;

(b) performing memory repair analysis of the defective memory cell information on a processing-unit-by-processing-unit basis to determine assignment of a spare line to the defective cell; and

(c) inserting a user function based on a user-defined defective memory cell repair condition between desired processing units used at step (b) to make a change to data processed by the memory repair analysis program.

BRIEF DESCRIPTION OF THE DRAWINGS

Fig. 1 is a conceptual block diagram of an embodiment of the present invention;

Fig. 2 is a block diagram showing a basic configuration of a general-purpose repair analysis part 32 in Fig. 1;

Fig. 3 is a conceptual diagram showing the relationship among an MRA program, MRA public functions, and a user analysis program;

Fig. 4 is a diagram for illustrating a naming rule for MRA public functions;

Fig. 5 is a diagram showing an exemplary user analysis program;

Fig. 6 is a functional block diagram showing another embodiment of the present invention;

Fig. 7 is a functional block diagram showing yet another embodiment of the present invention;

Fig. 8A shows a wafer on which semiconductor devices are formed;

Fig. 8B shows one block in a memory device;

Fig. 8C is a diagram for illustrating repair of defective memory cells;

Fig. 9 shows an example of a memory device having a redundancy structure;

Fig. 10 is a diagram illustrating a first condition of repair of the memory device shown in Fig. 9;

Fig. 11 is a diagram illustrating a second condition of repair of the memory device shown in Fig. 9;

Fig. 12 is a diagram illustrating a third condition of repair of the memory device shown in Fig. 9; and

Fig. 13 is a conceptual diagram illustrating an object to solve a problem with the conventional art.

BEST MODES FOR CARRYING OUT THE INVENTION

A mode for implementing present invention will be described with respect to embodiments thereof with reference to the accompanying drawings.

Fig. 1 shows a functional block diagram of an embodiment of the present invention. A semiconductor device test apparatus according to the present invention shown in Fig. 1 tests semiconductor devices having an memory in them (hereinafter simply referred to as memory devices), analyzes the location of a defect in a memory device if any, and determine an optimum solution as to which spare line (spare memory cell line) a defective line (memory cell column or row) of a memory device should be replaced with in order to repair the memory device. Only the essential elements of the apparatus will be described herein.

The semiconductor device test apparatus includes an engineering work station (EWS) 10, a test processor (TP) 20, and a repair analysis computing unit (RCPU) 30. Provided in the repair analysis computing unit 30 are fail memory 31 and a general-purpose repair analysis part 32. The engineering work station 10, test processor 20, and repair analysis computing unit 30 are capable of sending and receiving data, control signals, and programs between them through signal lines 15, 16, and 17. In addition, a test head 40 which electrically contacts a DUT to test the DUT is connected to the test processor 20 through a cable 18. The test head 40 provides a test signal from the test processor 20 to the DUT, receives a response signal from the DUT, and writes the result of the test in the fail memory unit 31. The general-purpose repair analysis part 32 analyzes the test result stored in the fail memory 31 to determine how a defective memory cell of the DUT should be repaired.

The engineering work station (EWS) 10, which is a computer used by

a measuring person for operating the apparatus, includes a repair condition file (RCF) storage 11 and a control part 12, sends out a control signal through the signal lines 15, 16, 17 under the control of the control part 12, downloads a program, and sends and receives data. Although not shown, the engineering work station 10 also includes input means for a user to input various settings and execution commands, and a display device as a GUI (Graphical User Interface) for displaying test processes and various kinds of data.

The test processor (TP) 20, which is a computer configured specially for the semiconductor device test apparatus, includes a test program storage 21 storing a test program for testing semiconductor devices, and performs control for function tests of DUTs. The test processor 20 executes a test program stored in the test program storage 21 to generate a test address, test data, and expectation value data, provides the test address and test data to the test head 40, and writes the test data in a memory cell in a memory device on the wafer attached to the test head 40 that is specified by the test address. The test processor 20 then compares data read from the address with the expectation data to determine whether the memory cell at the address is acceptable or defective. If defective, the test processor 20 writes data indicating the defect at the corresponding address in the fail memory 31 of the repair analysis computing unit 30. This test is performed on all addresses of the memory device and the test results are provided to the fail memory 31.

The general-purpose repair analysis part (general MRA) 32 of the repair analysis computing unit 30 includes an MRA program storage 32A, a user analysis program storage 32B, a analysis control part 32C, and data storage 32D, obtains required data from the fail memory 31 storing data on the results of tests on a DUT, executes an MRA program under the control of

the analysis control part 32C to perform repair analysis of defective cells, and determines the most efficient assignment of a spare line to the defective memory cells. The results of the analysis are sent to the test processor 20 for later use in physical repair of the DUT. The user analysis program storage 32B and the data storage 32D will be described with reference to Fig. 2.

Fig. 2 is a block diagram showing a basic functional configuration of the general-purpose repair analysis part 32 contained in the repair analysis computing unit 30 of Fig. 1. The general-purpose repair analysis part 32, which is an essential element of the present invention, includes the MRA program storage 32A storing a memory repair analysis program 32AP, the user analysis program storage 32B storing user analysis program functions 32BP, the analysis control part 32C, and the data storage 32D, as briefly described with reference to Fig. 1. The MRA program 32AP stored in the MRA program storage 32A is not a conventional program which merely provides models for memory repair solutions. Instead, an analysis process is split into parts for individual operations, and user function insertion points 32N1 - 32N5 are provided in the MRA program 32AP at which user functions 32B1 - 32B5 from the user analysis program 32BP can be inserted, each of which performs user-specific DUT-repair-analysis at the endpoint of the operation of the part. The data storage 32D stores data necessary for execution of the MRA program, for example repair condition files, data at intermediate stages of analysis, and analysis result data.

The MRA program 32AP includes: the step 32A1 of obtaining test result data from the fail memory 31 in response to a trigger indicating the completion of a test on a DUT from the test processor 20, for example, and initializing variables used in analysis; the step 32A2 of analyzing the test result data to assign a spare line for repairing a defective memory cell line; the

step 32A3 of assigning a spare line for repairing defective cells that remain after step 32A2; the step 32A4 of determining whether there is another way to assign a spare line after bit repair; and the step 32A5 of generating the results of analysis if there is not another way to assign a spare line. If another assigning way is found at step 32A4, spare line assignment is performed for bit fail repair at side step 32A3. These steps 32A1 - 32A5 are performed on each memory device. Information such as the memory size of the DUT, the number of blocks making up each block group, the number of row spare lines, and the number of column spare lines, is required in the line fail repair at step 32A2, the bit fail repair at step 32A3, and the determination at step 32A4. These items of information are loaded in the data storage 32D from a repair condition file RCF and used.

Splitting the analysis process in MRA program 32AP into parts 32 for individual operations refers to dividing the process into parts for operations such as acquisition of test result data and initialization of variables used in analysis (step 32A1), analysis of a line fail (step 32A2), analysis of a bit fail (step 32A3), and generation of the result of repair (step 32A5). Insertion points 32N1 - 32N5 are provided between these operations as necessary for providing and receiving user functions 32B1 - 32B5 so that individual user functions 32B1 - 32B5 making up a user-specific user analysis program 32BP can be inserted.

The user functions may include a function executed when initializing variables 32B1, a function executed after a line fail repair 32B2, a function executed after a bit fail repair 32B3, a function executed before generation of a result 32B4, and a function executed after the generation of the result 32B5. These functions are executed under the control of the analysis control part 32C. In particular, the function executed after a line fail repair checks repair

addresses to see whether there is identical repair addresses, for example. The function executed after a bit fail repair also checks repair addresses to see whether there is identical repair addresses. With the checking, appropriate spare lines to be used in repair of the same address can be selected.

In this way, a user can use a user function inserted to obtain analysis data, namely repair information and fail information, at a desired phase of processing according to the MRA program and thereby can repair a DUT having a special redundancy structure. In the embodiment described with respect to Figs. 1 and 2, the combination of the analysis control part 32C and the MRA program storage 32A constitutes memory repair analysis means that performs memory repair analysis of fail information; the combination of the analysis control part 32C and the user analysis program storage 32B constitutes user function means that inserts a user function between units of processing of an MAR program.

Fig. 3 schematically shows, as a variation of the embodiment shown in Figs. 1 and 2, the capability of exchanging information between the MRA program 32AP shown in Fig. 2 and a user function in the user analysis program 32BP through an MRA public function 32F that is directly intelligible to the user. If a user function 32B1 - 32B5 directly accesses the data storage 32D used by the MRA program 32AP to set data, data such as repair data could be corrupted by an incorrect value of the user function set by the user. Therefore, a function 32F called an MRA public function that acts as a filter is provided between the MRA program 32AP and the user function 32B1 - 32B5. By using the MRA public function 32F, a database can be referred to or modified safely within the user function 32B1 - 32B5. The user functions are written in C language in this embodiment.

Bearing in mind that the MRA public function 32F described with

respect to Fig. 3 is frequently used by engineers at manufactures and users of semiconductor device test apparatuses, a certain set of rules is defined so that users can readily write user analysis programs. MRA public function names are specified as shown in Fig. 4 in order to allow users to approximately understand the meaning of MRA public functions from their names, to minimize stress on the users during writing user functions, and to prevent the users from having impression that they are difficult to use.

The MRA public function name shown in Fig. 4 is an example that is written in accordance with certain rules, which will be described below. First, a tag, "Mra", is given to all functions. Then, a "class name" is assigned that indicates what type of data is dealt with. For example, the class name "Result" may be assigned to a function that deals with analysis result information stored in the repair analysis data storage 32D, the class name "Fail" to a function that deals with failure information, and the class name "Repair" to a function that deals with repair information.

A verb that indicates what the function does is then assigned. For example, the verb "Get" may be assigned to a function that gets information and the verb "Set" to a function that sets data. Finally, an object that indicates which information the function deals with is provided. As an example, a function name like "A = MraResultGetTotalBin" may be written. Various descriptions can be used since various rules can be established for writing programs. The essential thing is that anyone can easily write functions anytime. In this embodiment, a data check function portion 32FC is further provided that checks data set by a user function to see whether the data is valid or not. The data check function portion 32FC checks set data, for example an address value, the number of spare lines, a spare group number, or a block number inputted through a user function 32B1 - 32B5, to

see whether the data has an abnormal value, that is, whether the value is within a predetermined range. If an abnormal value is detected, a setting error is displayed on the engineering work station.

Fig. 5 shows an example of a user analysis program employing the description method shown in Fig. 4. The user analysis program 32BP is loaded from the engineering work station 10 into the user analysis program storage 32B. Lines 2 to 13 (UF1) are a user function that sorts a repair address portion; lines 14 to 29 (UF2) are a function that checks the repair addresses for the same address; lines 30 to 36 (UF3) are a main function of the user analysis program; and lines 37 to 44 (UF4) are a user setup function. Descriptions of MRA public functions can be seen on lines 8, 9, 24, and 25, for example.

For example in the MRA program 32AP shown in Fig. 2, the main function UF3 is executed at a user function insertion point 32N2. "SampleRuleCheck1" on the 30th line of the program in Fig. 5 represents a user function name. Accordingly, the check by "addressCheckRow" on line 34 is executed by the user function UF2 from line 14 to line 29. "MraPOINT_POST_REPAIR_LINEFAIL" on line 39 is the name assigned to the user function insertion point 32N2.

Line 20 in the user function UF2 uses the description method shown in Fig. 4 to represent an MRA public function, "MraBlockGroupGetRepairList". The number of block groups "blockGroupNo" and the row or column "dir" specified by the user in the list in the function indicate that the set values are checked for a predicted value when the function is performed, for example, to see whether "blockGroupNo" is less than or equal to 8 and whether "dir" is one of 1 (row) and 2 (column). If the user sets an invalid value, it is detected and an error message can be

displayed. This prevents an incorrect value from being set in the data storage 32D.

Fig. 6 shows a configuration of an embodiment in which MRA public functions shown in Figs. 4 and 5 are introduced. For simplicity, components such as a test processor 20 and test head 40 are not shown. In this embodiment, an MRA public function storage 23FC is provided in a general-purpose repair analysis part 32B in the configuration shown in Fig. 1. Before a test is started, an MRA program 32AP, a user analysis program 32BP, and MRA public functions 32F are loaded from an engineering work station 10 into an MRA program storage 32A, a user analysis program storage 32B, and an MRA public function storage 32FC, respectively. The combination of the analysis control part 32C and the MRA public function storage 32FC constitutes MRA public function means for inserting a user function between units of processing of the MRA program through an MRA public function.

Fig. 7 shows a configuration of the general-purpose repair analysis part 32 of an embodiment in which user functions can be selected to be used for tests on different types of DUTs in the embodiment shown in Fig. 1. Different repair condition files RCFs are provided for different types of semiconductor devices. Accordingly, each time a different type of DUT is selected, the RCF corresponding to that type of DUT must be selected before an MRA program is executed. If changeover from one type of DUT to another is made frequently, it is time consuming and inefficient to load an RCF from the engineering work station 10 to the general-purpose repair analysis part 32 each time changeover is made. Therefore, sets of user functions 32BJ and 32BK that correspond to different types of DUTs are provided in a user analysis program 32BP and are stored in the user analysis program storage 32B so that the user functions 32BJ, 32BK can be readily

selected in accordance with the type of DUT. Rule names "J" and "K" are assigned to the sets of user functions 32BJ and 32BK, respectively.

Further provided in the general-purpose repair analysis part 32 is an RCF storage 32F, in which a repair condition file RCF1 with the rule name "none" that is independent of the type of DUT, and repair condition files RCF2 and RCF3 with the rule names "J" and "K" that are dependent on the type of DUT are provided. The RCF with the rule name "J" is used for memory devices having a capacity of 128 Mbytes and the RCF with the rule name "K" is used for memory devices having a capacity of 256 Mbytes. Because the rule names also function as identities of DUTs, a user can easily define user functions.

The repair condition files (RCFs) are stored in the repair condition file storage 11 of the engineering work station 10 as described above and repair analysis is performed based on these files. By specifying a rule name in an RCF, user functions can be changed readily in accordance with the type of DUT at run-time.

If no rule name is specified in an RCF, the MRA program 32AP performs original repair analysis. If a rule name is specified in an RCF as shown in Fig. 7, the set of user functions 32BK specified for the rule name "K" is executed. For example, "SAMPLE_RULE_CHECK1" on lines 39 and 41 corresponds to the rule name "J" of repair condition file RCF2 (or "K" of RCF3) in the user analysis program of Fig. 5.

In this way, user functions intended for a DUT can be readily selected on a single semiconductor device test apparatus that is used for testing different types of DUTs.

EFFECTS OF THE INVENTION

As has been stated earlier, the conventional MRAs have been

sufficiently effective for repair of conventional memory ICs. However, it has become difficult to repair DUTs containing memory devices having special, user-specific redundancy structures with the conventional MRAs in recent years, which has led to yield reduction.

According to the present invention, a general-purpose repair analysis part 32 is provided in which a newly provided MRA program 32AP and a user analysis program 32BP are caused to work with each other so that DUTs having special redundancy structures can be tested. As a result, the need for the on-wafer post process is eliminated. Furthermore, because user functions that represent user-specific operations can be incorporated, repair analysis can be retried until a user-specific operation is found to be able to repair.

Because the user-specific operations run on a repair analysis computing unit (RCPU) 30, a function test on a DUT by a test processor 20 can be performed in parallel with memory repair analysis by a repair analysis computing unit 30 when multiple DUTs are tested at a time. As a result, the run time can be significantly reduced.

Thus, the present invention enables near complete repair analysis of DUTs containing a semiconductor memory device and has a great technical effect.